Name of Presentation

Team: Simulacrum

Members: Ethan Urie, Swaroop Choudhari, Yudi Nagata, Zach Mouri, Saul Jaspan

Project Presentation

17-654: Analysis of Software Artifacts

Agenda

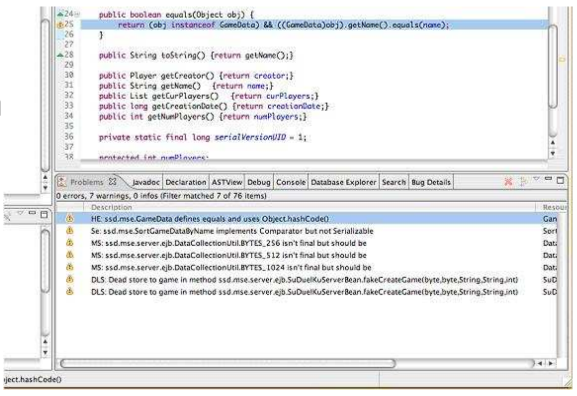
- The Tool
- The Tests
- The Results
- Lessons Learned

The Tool

- Find Bugs
  - http://findbugs.sourceforge.net/
  - Written in Java
  - Open source
  - Developed by University of Maryland
  - Does static analysis of Java classes
  - Uses BCEL
  - Uses “Bug patterns”

The Tool

- Accessible through
  - CLI
  - Swing application
  - Eclipse plug-in



```
public boolean equals(Object obj) {
    return (obj instanceof GameData) && ((GameData)obj).getName().equals(name);
}

public String toString() {return getName();}

public Player getCreator() {return creator;}
public String getName()   {return name;}
public List getCurPlayers()   {return curPlayers;}
public long getCreationDate() {return creationDate;}
public int getNumPlayers() {return numPlayers;}

private static final long serialVersionUID = 1;

protected int numPlayers;
```

```
0 errors, 7 warnings, 0 infos (Filter matched 7 of 76 items)
HE: ssd.mse.GameData defines equals and uses Object.hashCode()
Se: ssd.mse.SortGameDataByName implements Comparator but not Serializable
MS: ssd.mse.server.ejb.DataCollectionUtil.BYTES_256 isn't final but should be
MS: ssd.mse.server.ejb.DataCollectionUtil.BYTES_512 isn't final but should be
MS: ssd.mse.server.ejb.DataCollectionUtil.BYTES_1024 isn't final but should be
DLS: Dead store to game in method ssd.mse.server.ejb.SuDuelKuServerBean.fakeCreateGame(byte,byte,String,String,int)
DLS: Dead store to game in method ssd.mse.server.ejb.SuDuelKuServerBean.fakeCreateGame(byte,byte,String,String,int)
ject.hashCode()
```

BECL

- Byte Code Engineering Library (BECL, http://jakarta.apache.org/bcel)
- Parses Java byte code
- Classes are represented by objects
  - Contain symbolic info of class
  - Methods
  - Fields
  - Etc.

FindBugs

- Looks for bug patterns
  - A code idiom that is likely to be an error
- Can easily detect these with simple static analysis
- Framework that can do
  - Class structure analysis
  - Linear code scans
  - Control sensitive analysis
  - Dataflow analysis

Example Bug Patterns

- Suspicious equals comparison
- Equal objects must have equal hash codes
- Inconsistent synchronization
- Non-serializable Serializable class
- Return value should be checked

Experiment Setup

- Team Bots
  - Open source API for intelligent mobile agents
  - 20 KLOC
  - 231 classes
- SuDuelKu
  - EJB multi-player SuDuKu game
  - 7 KLOC
  - 183 classes

FindBugs Setup

- Max effort
- Medium priority
- Look for
  - Correctness
  - Multithreaded correctness
  - Performance

What we did

- For each “bug” reported by FindBugs we
  - Validated bug
  - Measured validation time
  - Fixed the bug
  - Measured fix time

Results

- Total Bugs: 47
- False positives: 12
- Bugs: 35
- Average time to verify: 1.5 mins
- Average time to fix: 2.5 mins
- Runtime - Teambots: 26 seconds
- Runtime - SuDuelKu: 15 seconds

Lessons Learned

- Many bugs can be found using bug patterns
- Not many false positives
- Lots of faults, not many errors
- Some pattern detectors are very accurate, others are not
- Bugs found were simple to validate and fix

Questions

Questions?

EclipsePro

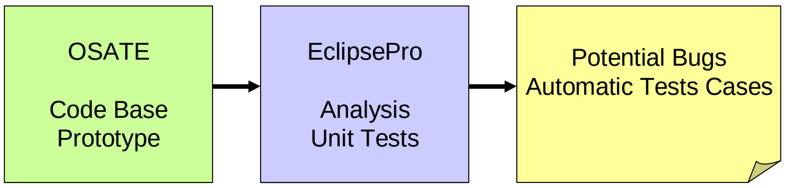
Team DaVinci

Christopher Nelson
Luis Rios
Chung-Hao Shih

17-654: Analysis of Software Artifacts

EclipsePro

- Provides analysis of code, automatic test cases, and test coverage reports
- Goals of the evaluation



EclipsePro - Analysis

Looking into Performance

Issue	Detected
Append string instead of char	9
Concatenation in appending method	2
Initial capacity for collections	22
Method invocation in loop condition	10
Variable declared within loop	339

EclipsePro - Analysis

Looking into Maintainability

Issue	Detected
Exceeded length of methods	32
Empty methods	53
Exceptions with no logging	10
String literals	1970
Unused methods	2

EclipsePro - Analysis

- Benefits
  - Configurable rules for detecting issues
  - Processing time is good
  - Integrates in one tool
    - Analysis of Source Code
    - Generation of Unit Tests
    - Analysis of Code Coverage
    - Metrics of Source Code

EclipsePro - Analysis

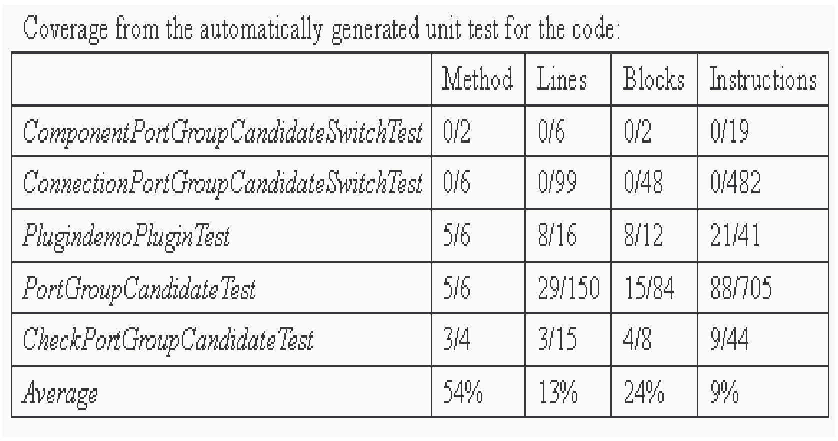
- Drawbacks
  - False positives for unused fields
  - Do not detect that some variables must be declared as constants
  - Constant conditional expressions such as while(true) are always reported
  - Hiding inherited fields does not allow to ignore special fields such as copyright notices

EclipsePro – Test Cases

- Test Environment
  - OSATE Libraries
  - EclipsePro
  - Prototype
    - *ComponentPortGroupCandidateSwitch.java*
    - *ConnectionPortGroupCandidateSwitch.java*
    - *PlugindemoPlugin.java*
    - *PortGroupCandidate.java*
    - *CheckPortGroupCandidate.java*

EclipsePro – Test Cases

Test Coverage Report



Coverage from the automatically generated unit test for the code:

	Method	Lines	Blocks	Instructions
ComponentPortGroupCandidateSwitchTest	0/2	0/6	0/2	0/19
ConnectionPortGroupCandidateSwitchTest	0/6	0/99	0/48	0/482
PlugindemoPluginTest	5/6	8/16	8/12	21/41
PortGroupCandidateTest	5/6	29/150	15/84	88/705
CheckPortGroupCandidateTest	3/4	3/15	4/8	9/44
Average	54%	13%	24%	9%

EclipsePro – Test Cases

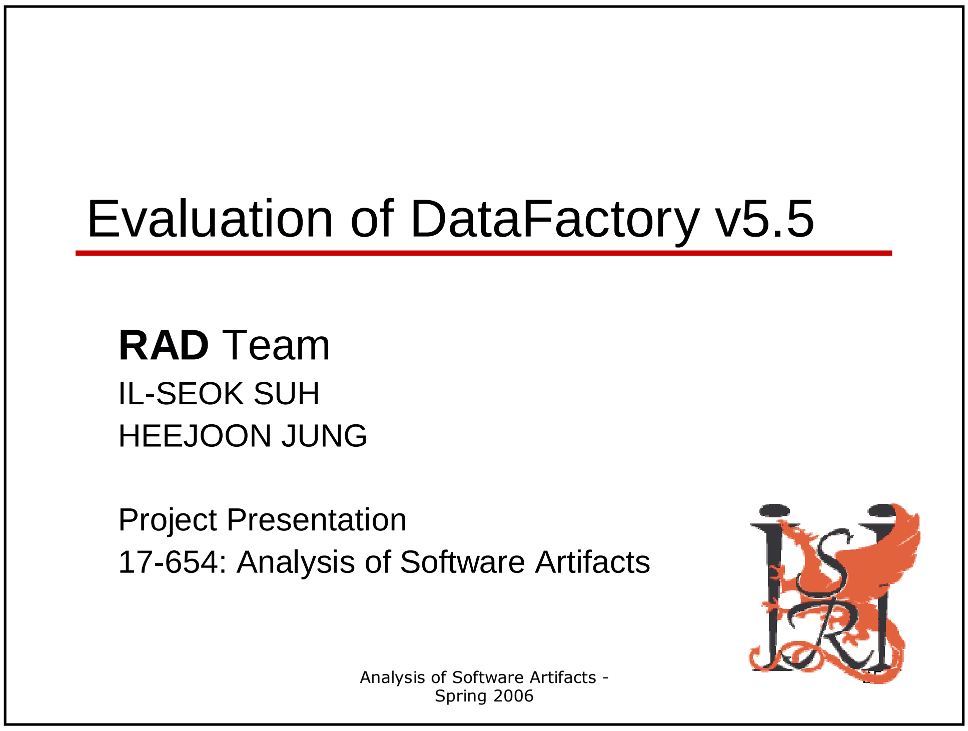
- Benefits
  - Generate the framework for unit tests
  - Generate the basic test methods
  - Generate regression test cases
  - Provide mechanism for human recheck
  - Check valid and invalid parameters for each method call
  - Generate comments
  - Provide test coverage report
  - Easy to use
  - Save time with automatic test generation

EclipsePro – Test Cases

- Drawbacks
  - Do not work with interfaces
  - Miss libraries from original source code
  - Miss other basic unit test scenarios
  - Have low test coverage
  - Still need human interaction

EclipsePro

- Questions



Evaluation of DataFactory v5.5

RAD Team
IL-SEOK SUH
HEEJOON JUNG

Project Presentation
17-654: Analysis of Software Artifacts

Analysis of Software Artifacts -
Spring 2006

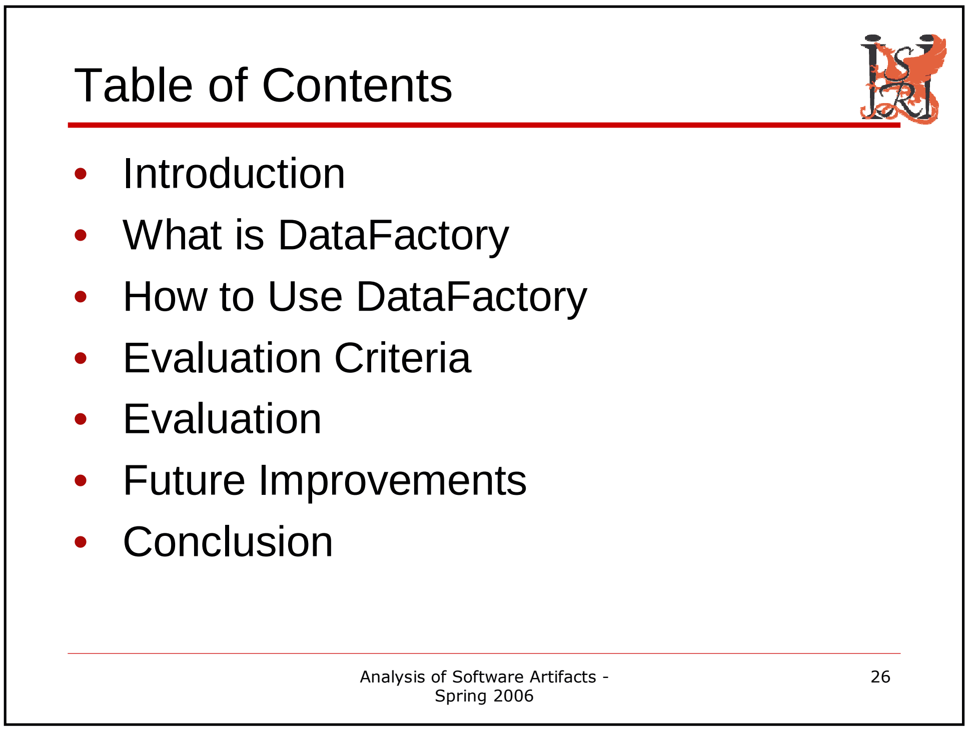


Table of Contents

Introduction

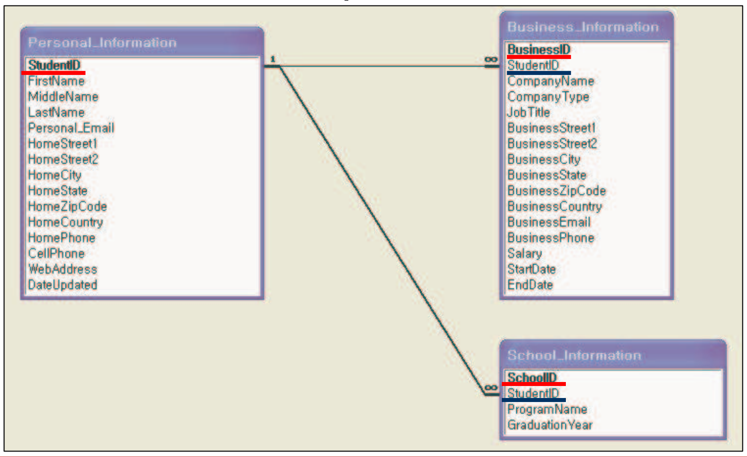
- Practicum Project
  - The team is going on a practicum project: Re-engineering of MSE and MSIT-SE Alumni Database
- Purpose
  - Need to use test data for the practicum project
- Expectation
  - By using a test data generator tool, the team will be able to get useful test cases and test database redesigned
- Approach
  - Make evaluation criteria
  - Redesign tables in the database
  - Make a connection between the tool and the database
  - Execute the tool using the inputted tables
  - Evaluate the tool according to the criteria

What is DataFactory

- A test generator tool developed by Quest Software
  - Load a schema from database
  - Display database tables and fields
  - Produce meaningful test data
  - Write the test data to output files or save into the database
- Fast and easy way to generate test cases

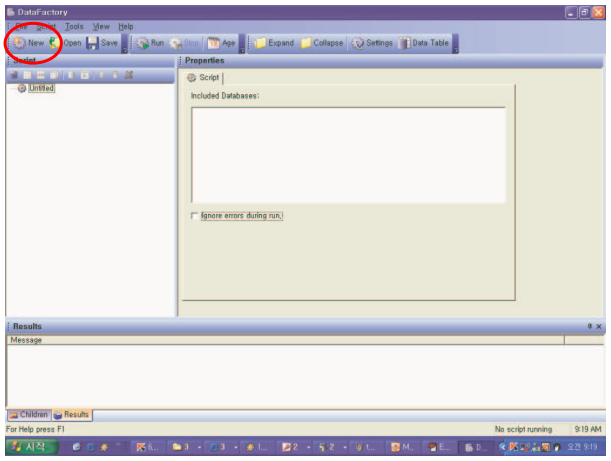
How to Use DataFactory

- Table Relationship



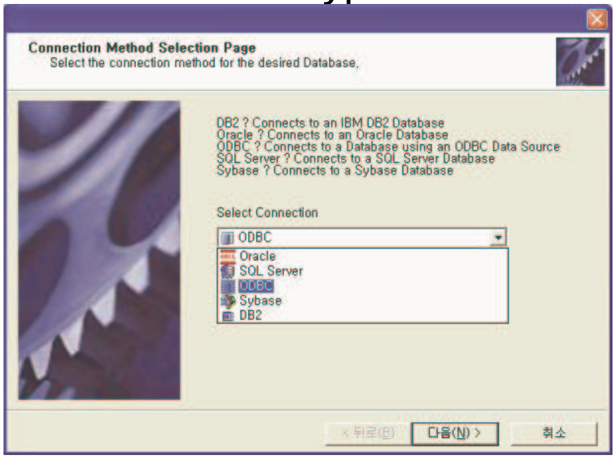
How to Use DataFactory

- Make new connection with database



How to Use DataFactory

- Select connection type


Connection Method Selection Page
Select the connection method for the desired Database.

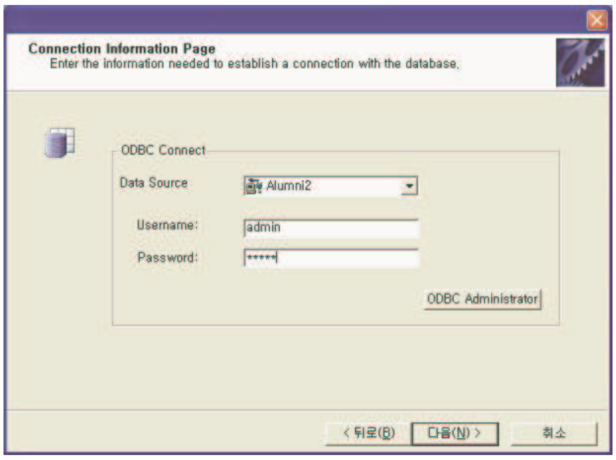
DB2 ? Connects to an IBM DB2 Database
Oracle ? Connects to an Oracle Database
ODBC ? Connects to a Database using an ODBC Data Source
SQL Server ? Connects to a SQL Server Database
Sybase ? Connects to a Sybase Database

Select Connection: ODBC

Oracle
SQL Server
ODBC
Sybase
DB2
< 뒤로(B)
다음(N) >
취소

How to Use DataFactory

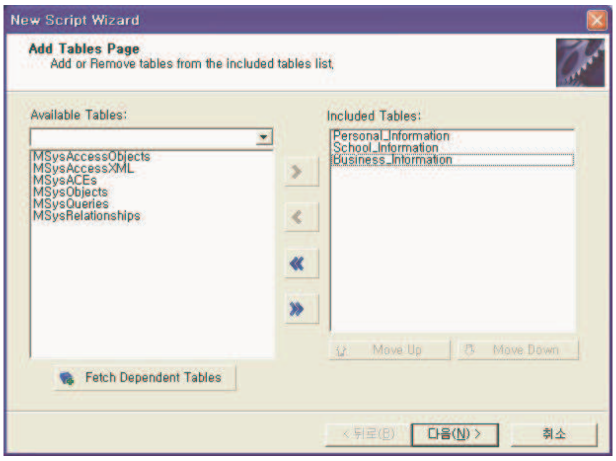
- Select ODBC


Connection Information Page
Enter the information needed to establish a connection with the database.
ODBC Connect
ODBC Administrator
< 뒤로(B)
다음(N) >
취소

Data Source: Alumni2
Username: admin
Password: *****

How to Use DataFactory

- Select tables



New Script Wizard

Add Tables Page
Add or Remove tables from the included tables list.

Available Tables:

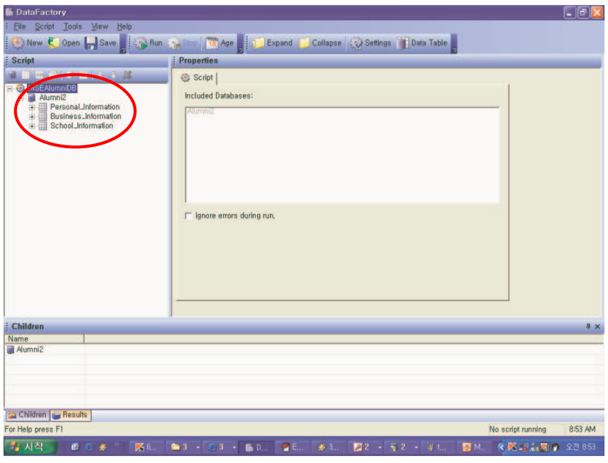
MSysAccessObjects
MSysAccessXML
MSysACEs
MSysObjects
MSysQueries
MSysRelationships

Included Tables:

Personal_Information
School_Information
Business_Information

How to Use DataFactory

- After connection



DataFactory

Script

ISEAlumniDB
Alumni2
Personal_Information
Business_Information
School_Information

Properties

Included Databases:
Alumni2

☐ Ignore errors during run.

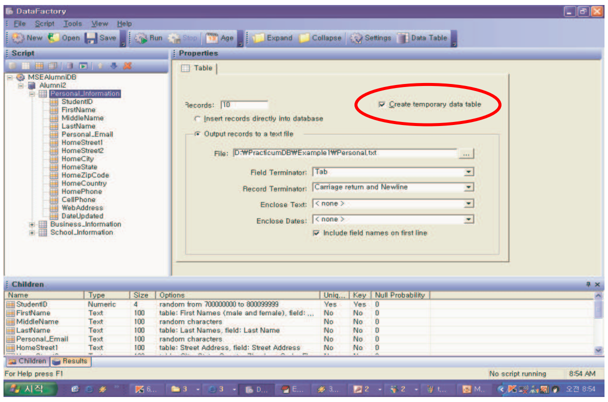
Children

Name
Alumni2

For Help press F1 — No script running — 8:53 AM

How to Use DataFactory

- Check the “Create Temporary Data Table” for data integrity


Records: 10

[✓] Create temporary data table

( ) Insert records directly into database

(•) Output records to a text file

File: D:₩PracticumDB₩Example1₩Personal.txt

Field Terminator: Tab

Record Terminator: Carriage return and Newline

Enclose Text: < none >

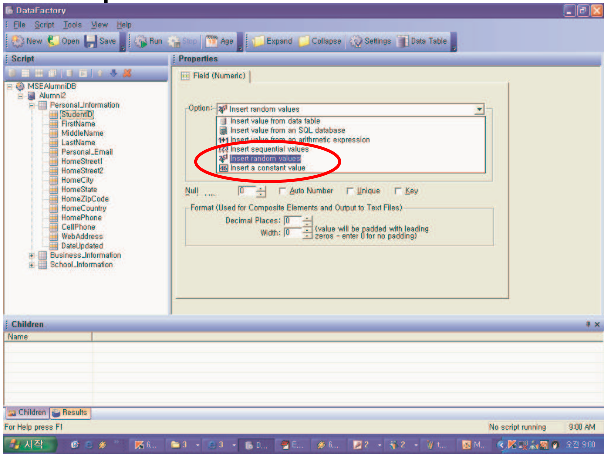
Enclose Dates: < none >

[✓] Include field names on first line

Name	Type	Size	Options	Uniq...	Key	Null Probability
StudentID	Numeric	4	random from 700000000 to 800099999	Yes	Yes	0
FirstName	Text	100	table: First Names (male and female), field: ...	No	No	0
MiddleName	Text	100	random characters	No	No	0
LastName	Text	100	table: Last Names, field: Last Name	No	No	0
Personal_Email	Text	100	random characters	No	No	0
HomeStreet1	Text	100	table: Street Address, field: Street Address	No	No	0

How to Use DataFactory

- Select option


Field (Numeric)

Option: Insert random values

- Insert value from data table
- Insert value from an SQL database
- Insert value from an arithmetic expression
- Insert sequential values
- [✓ Insert random values]
- Insert a constant value

Null 0 [ ] Auto Number [ ] Unique [ ] Key

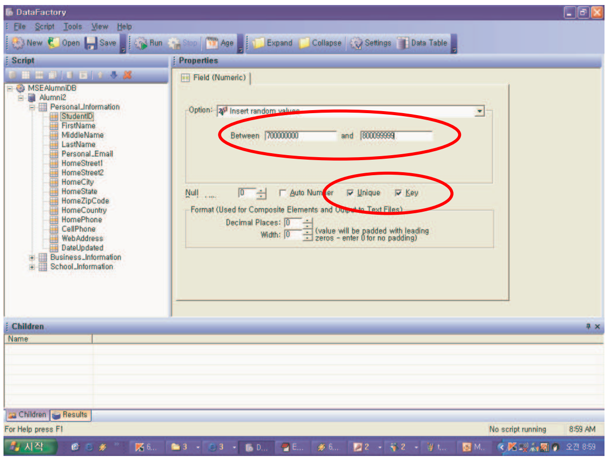
Format (Used for Composite Elements and Output to Text Files)

Decimal Places: 0

Width: 0 (value will be padded with leading zeros - enter 0 for no padding)

How to Use DataFactory

- Enter condition



Field (Numeric)

Option: Insert random values

Between 700000000 and 800099999

Null 0 ☐ Auto Number ☑ Unique ☑ Key

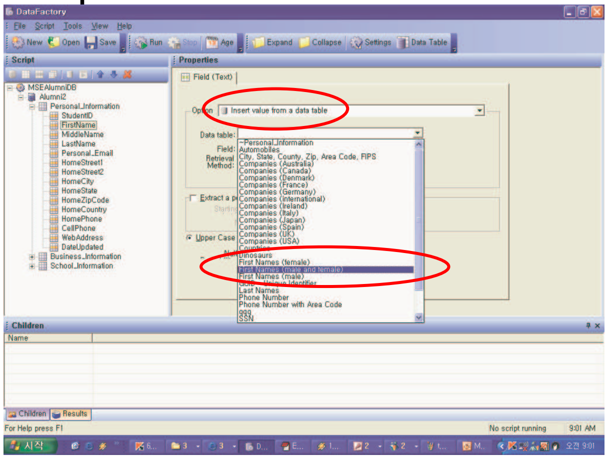
Format (Used for Composite Elements and Output to Text Files)

Decimal Places: 0

Width: 0 (value will be padded with leading zeros - enter 0 for no padding)

How to Use DataFactory

- Select predefined data



Field (Text)

Option: Insert value from a data table

Data table:

Field:

Retrieval Method:

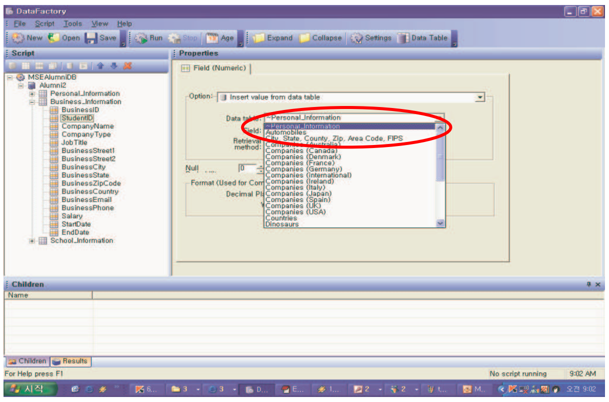
☐ Extract a p

Upper Case

~Personal_Information
Automobiles
City, State, County, Zip, Area Code, FIPS
Companies (Australia)
Companies (Canada)
Companies (Denmark)
Companies (France)
Companies (Germany)
Companies (international)
Companies (Ireland)
Companies (Italy)
Companies (Japan)
Companies (Spain)
Companies (UK)
Companies (USA)
Countries
Dinosaurs
First Names (female)
First Names (male and female)
First Names (male)
GUID - Unique Identifier
Last Names
Phone Number
Phone Number with Area Code
ggg
SSN

How to Use DataFactory

- Select Personal_Information table to maintain data integrity


DataFactory
File Script Tools View Help
New Open Save Run Stop Age Expand Collapse Settings Data Table
Script
Properties
Children
Name
Children Results

MSEAlumniDB
Alumni2
Personal_Information
Business_Information
BusinessID
StudentID
CompanyName
CompanyType
JobTitle
BusinessStreet1
BusinessStreet2
BusinessCity
BusinessState
BusinessZipCode
BusinessCountry
BusinessEmail
BusinessPhone
Salary
StartDate
EndDate
School_Information

Field (Numeric)

Option: Insert value from data table

Data table: ~Personal_Information

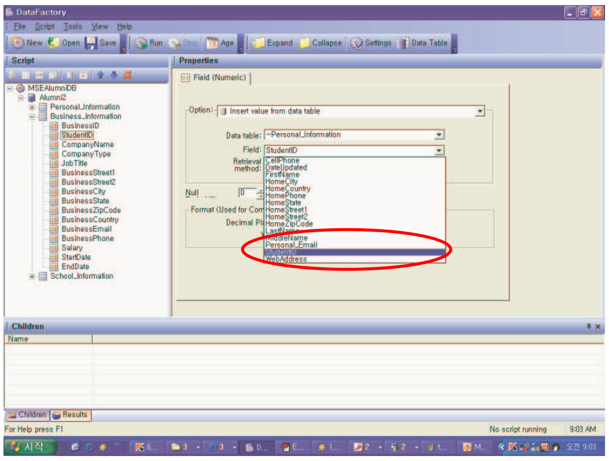
~Personal_Information
Automobiles
City, State, County, Zip, Area Code, FIPS
Companies (Australia)
Companies (Canada)
Companies (Denmark)
Companies (France)
Companies (Germany)
Companies (International)
Companies (Ireland)
Companies (Italy)
Companies (Japan)
Companies (Spain)
Companies (UK)
Companies (USA)
Countries
Dinosaurs

Field:
Retrieval method:
Null 0
Format (Used for Com
Decimal Pl

For Help press F1 No script running 9:02 AM

How to Use DataFactory

- Select StudetID field for referential key


DataFactory
File Script Tools View Help
New Open Save Run Stop Age Expand Collapse Settings Data Table
Script
Properties
Children
Name
Children Results

MSEAlumniDB
Alumni2
Personal_Information
Business_Information
BusinessID
StudentID
CompanyName
CompanyType
JobTitle
BusinessStreet1
BusinessStreet2
BusinessCity
BusinessState
BusinessZipCode
BusinessCountry
BusinessEmail
BusinessPhone
Salary
StartDate
EndDate
School_Information

Field (Numeric)

Option: Insert value from data table

Data table: ~Personal_Information

Field: StudentID

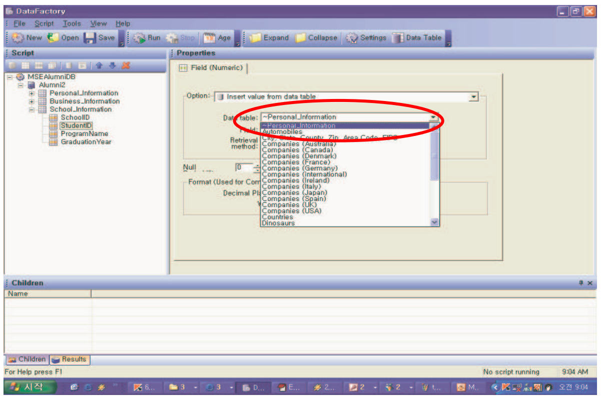
CellPhone
DateUpdated
FirstName
HomeCity
HomeCountry
HomePhone
HomeState
HomeStreet1
HomeStreet2
HomeZipCode
LastName
MiddleName
Personal_Email
StudentID
WebAddress

Retrieval method:
Null 0
Format (Used for Com
Decimal Pl

For Help press F1 No script running 9:03 AM

How to Use DataFactory

- Select Personal_Information table to maintain data integrity



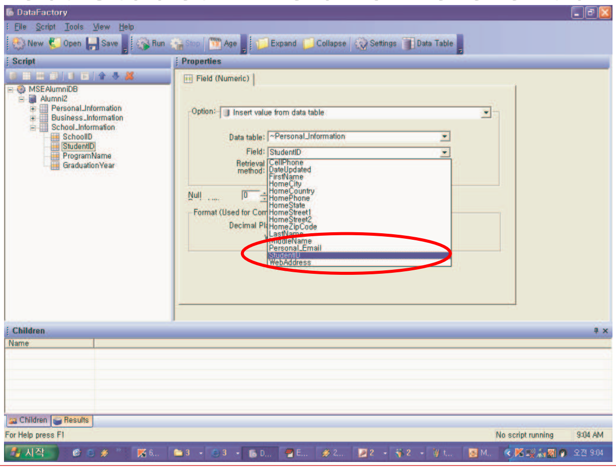
Option: Insert value from data table

Data table: ~Personal_Information

~Personal_Information
Automobiles
City, State, County, Zip, Area Code, FIPS
Companies (Australia)
Companies (Canada)
Companies (Denmark)
Companies (France)
Companies (Germany)
Companies (international)
Companies (Ireland)
Companies (Italy)
Companies (Japan)
Companies (Spain)
Companies (UK)
Companies (USA)
Countries
Dinosaurs

How to Use DataFactory

- Select StudetID field for referential key



Option: Insert value from data table

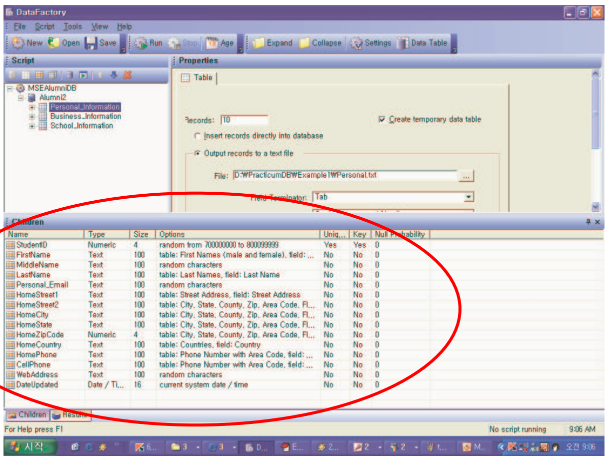
Data table: ~Personal_Information

Field: StudentID

CellPhone
DateUpdated
FirstName
HomeCity
HomeCountry
HomePhone
HomeState
HomeStreet1
HomeStreet2
HomeZipCode
LastName
MiddleName
Personal_Email
StudentID
WebAddress

How to Use DataFactory

- Check the conditions



Records: 10

☑ Create temporary data table

○ Insert records directly into database

● Output records to a text file

File: D:₩PracticumDB₩Example1₩Personal.txt

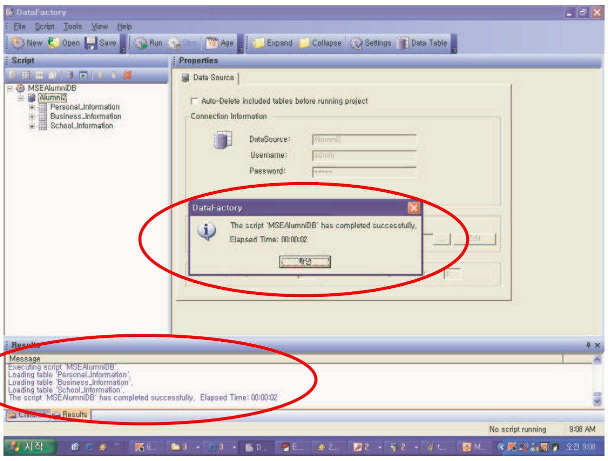
Field Terminator: Tab

Children

Name	Type	Size	Options	Uniq...	Key	Null Probability
StudentID	Numeric	4	random from 700000000 to 800099999	Yes	Yes	0
FirstName	Text	100	table: First Names (male and female), field: ...	No	No	0
MiddleName	Text	100	random characters	No	No	0
LastName	Text	100	table: Last Names, field: Last Name	No	No	0
Personal_Email	Text	100	random characters	No	No	0
HomeStreet1	Text	100	table: Street Address, field: Street Address	No	No	0
HomeStreet2	Text	100	table: City, State, County, Zip, Area Code, Fl...	No	No	0
HomeCity	Text	100	table: City, State, County, Zip, Area Code, Fl...	No	No	0
HomeState	Text	100	table: City, State, County, Zip, Area Code, Fl...	No	No	0
HomeZipCode	Numeric	4	table: City, State, County, Zip, Area Code, Fl...	No	No	0
HomeCountry	Text	100	table: Countries, field: Country	No	No	0
HomePhone	Text	100	table: Phone Number with Area Code, field: ...	No	No	0
CellPhone	Text	100	table: Phone Number with Area Code, field: ...	No	No	0
WebAddress	Text	100	random characters	No	No	0
DateUpdated	Date / Ti...	16	current system date / time	No	No	0

How to Use DataFactory

- Result



☐ Auto-Delete included tables before running project

Connection Information

DataSource: Alumni2

Username: admin

Password: *****

DataFactory

The script 'MSEAlumniDB' has completed successfully.

Elapsed Time: 00:00:02

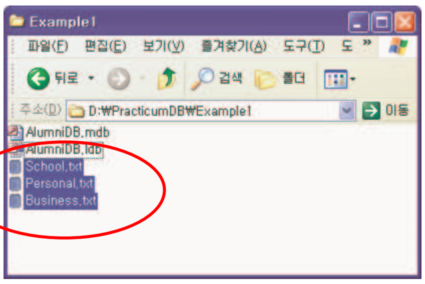
확인

Results

```
Message
Executing script 'MSEAlumniDB'.
Loading table 'Personal_Information'.
Loading table 'Business_Information'.
Loading table 'School_Information'.
The script 'MSEAlumniDB' has completed successfully.  Elapsed Time: 00:00:02
```

How to Use DataFactory

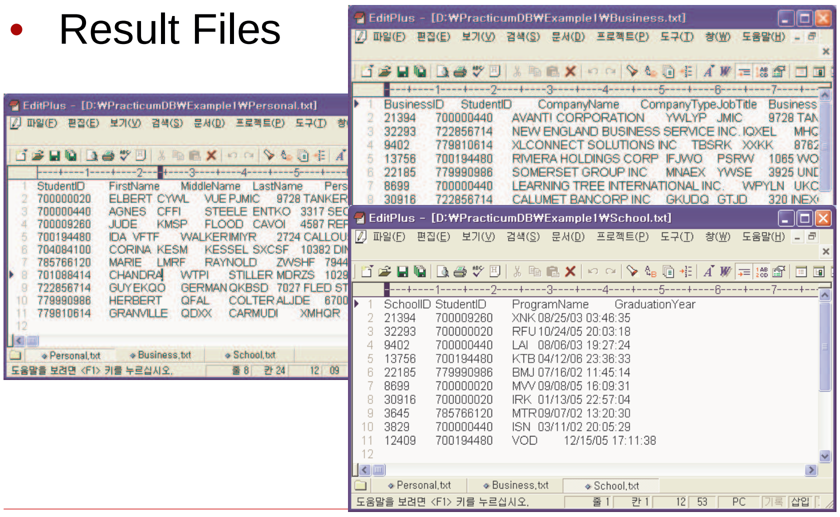
- New test files



```
D:₩PracticumDB₩Example1
AlumniDB.mdb
AlumniDB.ldb
School.txt
Personal.txt
Business.txt
```

How to Use DataFactory

- Result Files


파일(F) 편집(E) 보기(V) 검색(S) 문서(D) 프로젝트(P) 도구(T) 창(W) 도움말(H)
Personal.txt
Business.txt
School.txt
도움말을 보려면 <F1> 키를 누르십시오.

EditPlus - [D:₩PracticumDB₩Example1₩Personal.txt]

```
StudentID    FirstName    MiddleName   LastName    Pers
700000020    ELBERT CYWL    VUE PJMIC    9728 TANKER
700000440    AGNES   CFFI    STEELE  ENTKO   3317 SEC
700009260    JUDE    KMSP    FLOOD   CAVOI   4587 REF
700194480    IDA  VFTF    WALKER IMIYR    2724 CALLOU
704084100    CORINA  KESM    KESSEL SXCSF    10382 DIN
785766120    MARIE   LMRF    RAYNOLD     ZWSHF   7944
701088414    CHANDRA    WTPI    STILLER MDRZS   1029
722856714    GUY EKQO    GERMAN QKBSD    7027 FLED ST
779990986    HERBERT    QFAL    COLTER ALJDE    6700
779810614    GRANVILLE    QDXX    CARMUDI     XMHQR
```

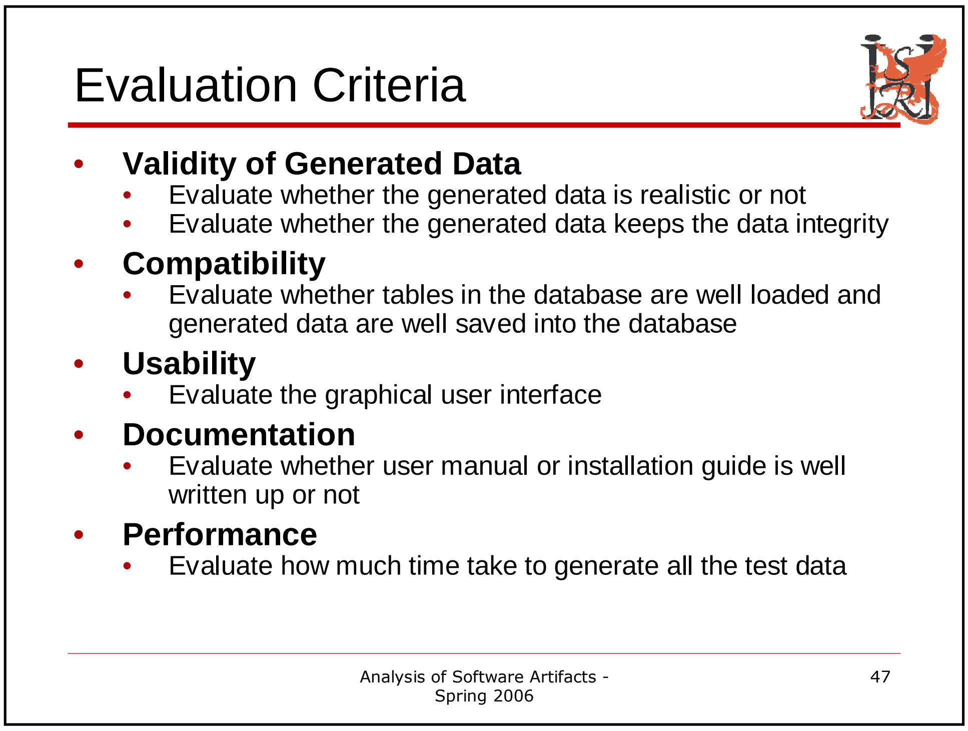
EditPlus - [D:₩PracticumDB₩Example1₩Business.txt]

```
BusinessID   StudentID    CompanyName    CompanyTypeJobTitle   Business
21394    700000440    AVANTI CORPORATION    YWLYP   JMIC    9728 TAN
32293    722856714    NEW ENGLAND BUSINESS SERVICE INC. IQXEL    MHC
9402     779810614    XLCONNECT SOLUTIONS INC    TBSRK   XXKK    8762
13756    700194480    RIVIERA HOLDINGS CORP  IFJWO    PSRW    1065 WO
22185    779990986    SOMERSET GROUP INC    MNAEX   YWSE    3925 UND
8699     700000440    LEARNING TREE INTERNATIONAL INC.    WPYLN   UKC
30916    722856714    CALUMET BANCORP INC    GKUDQ  GTJD    320 INEX
```

EditPlus - [D:₩PracticumDB₩Example1₩School.txt]

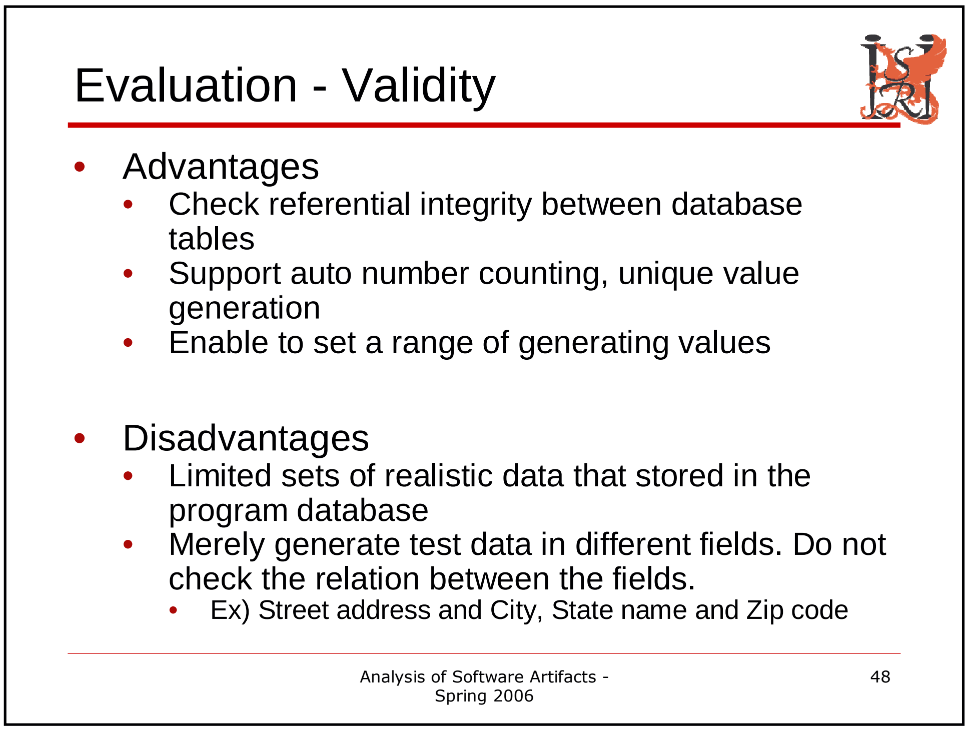
```
SchoolID StudentID    ProgramName    GraduationYear
21394    700009260    XNK 08/25/03 03:46:35
32293    700000020    RFU 10/24/05 20:03:18
9402     700000440    LAI   08/06/03 19:27:24
13756    700194480    KTB 04/12/06 23:36:33
22185    779990986    BMJ 07/16/02 11:45:14
8699     700000020    MVV 09/08/05 16:09:31
30916    700000020    IRK  01/13/05 22:57:04
3645     785766120    MTR 09/07/02 13:20:30
3829     700000440    ISN  03/11/02 20:05:29
12409    700194480    VOD       12/15/05 17:11:38
```

```
줄 8   칸 24   12   09
줄 1   칸 1   12   53   PC   기록   삽입
```



Evaluation Criteria

- **Validity of Generated Data**
  - Evaluate whether the generated data is realistic or not
  - Evaluate whether the generated data keeps the data integrity
- **Compatibility**
  - Evaluate whether tables in the database are well loaded and generated data are well saved into the database
- **Usability**
  - Evaluate the graphical user interface
- **Documentation**
  - Evaluate whether user manual or installation guide is well written up or not
- **Performance**
  - Evaluate how much time take to generate all the test data



Evaluation - Validity

- Advantages
  - Check referential integrity between database tables
  - Support auto number counting, unique value generation
  - Enable to set a range of generating values
- Disadvantages
  - Limited sets of realistic data that stored in the program database
  - Merely generate test data in different fields. Do not check the relation between the fields.
    - Ex) Street address and City, State name and Zip code

Evaluation - Compatibility

- Advantages
  - Enable to direct access various DBMS and ODBC compliant database
    - Ex) Oracle, DB2, SQL server, and Sybase
- Disadvantages
  - Incomplete compatibility with ODBC compliant database
    - Error occurs when test data are saved
  - Do not support all the major DBMS
    - Ex) DataFactory does not support direct access to FileMaker, so it should pass through ODBC
  - Once tables are loaded from database, relations of the tables in the database are not maintained
    - Additional setting up is required

Evaluation - Usability

- Advantages
  - Provide simple and plain graphical user interface
  - Provide "Children View"
    - Easy to recognize which field attributes are set up
- Disadvantages
  - Do not have its own viewer to display outputs
    - Hard to read output data
  - Should set up additional items for checking referential integrity
  - Do not know the relationships between tables
  - Should have database and DBMS to run the system
    - Impossible to simply generate test data without database

Evaluation - Documentation

- Definitely insufficient documentation. Very limited resources
- Advantages
  - Provide simple tutorials
- Disadvantages
  - There is no official documentation
  - Do not provide installation guide
  - Do not provide user manual
    - Hard to know system functionalities or how to use
  - Do not provide error lists or exception lists

Evaluation - Performance

- Measured by elapsed time
- Performance would be lowered if the number of tables in the database are increase and the number of fields are increased

Future Improvements

- Make an official documentation
  - Need to make user manual
- Support direct access to more DBMS
  - Should be compatible with FileMaker Pro v. 8.0
- Update graphical user interface
  - Hard to recognize current output results
- Should be a standalone application that does not require database
- Improve the validity of the generated data

Analysis Application of Purify

- Utilization of Purify in the Navigation Data Converter Application

Pathfinder

Jihye Eom, Wangbong Lee, Youngseok Oh

Project Presentation

17-654: Analysis of Software Artifacts

Contents

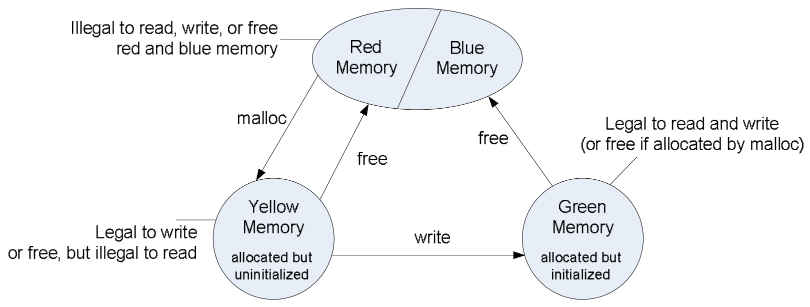
- Purify
- How Purify Works
- Memory State in Purify
- Purify for Java
- Project Introduction
- How to Apply Purify
- Application work
- Benefit & Drawback of Purify

Purify

- IBM Rational Purify
  - Automatic error detection tool
    - Finding runtime errors
    - Finding memory leak
- Working Environments
  - OS
    - Windows, Linux, Unix
  - Language support
    - C/C++, Java
    - C#, VB.NET in Windows

How Purify Works

- Tracking the status of memory used by program


Illegal to read, write, or free red and blue memory
Red Memory
Blue Memory
malloc
free
free
Legal to read and write (or free if allocated by malloc)
Yellow Memory
allocated but uninitialized
Legal to write or free, but illegal to read
write
Green Memory
allocated but initialized

Memory State in Purify

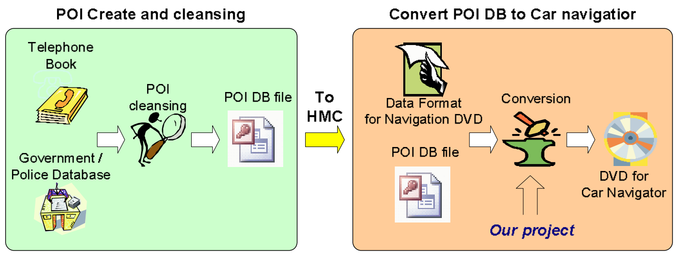
- Red Memory
  - Purify labels heap and stack memory “red” initially.
  - Unallocated and Freed uninitialized memory
  - Illegal OP: read, write, free
  - Not owned by the program
- Yellow Memory
  - Memory returned by new and malloc
  - It has been allocated, but uninitialized
  - You can write, and free (if allocated by malloc)
  - Illegal OP: read
- Green Memory
  - Allocated memory and written memory
  - You can read, write, and free (if allocated by malloc)
- Blue Memory
  - Freed initialized memory
  - Illegal OP: read, write, free

Purify for Java

- Garbage Collector (GC)
  - JVM uses garbage collection to collect unused memory automatically
  - GC won't be automatically run until a program needs more memory than is currently available
  - When GC is missed possibly
    - Resetting the reference to another object
    - Changing the state of an object when there is still a reference to the old state
    - Having a reference that is pinned by a long running thread
- Memory Leak
  - The memory garbage occupied by the object that would not be referred any more
  - More and more, becomes big one
  - HELP ME, Purify!

Studio Project

Navigation Data Production Process



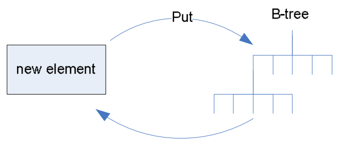
- Design / Implement converter (converting POI DB to navigation data)
- Tree-Structured Index File
  - For fast search not using DBMS
  - For manipulate large amount of POI Data in disk media
  - Tree algorithm is necessary

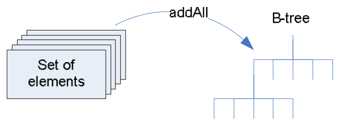
How to Apply Purify

- Run different B-tree source code
  - Choose better one in terms of performance
  - See the overall memory, memory profile, execution speed
- Tweak the performance
  - Compare Memory / Execution speed before and after

Source Introduction

- Source A
  - Put elements one by one
- Source B
  - Put elements in Vector form
- Different insert way to put element into tree

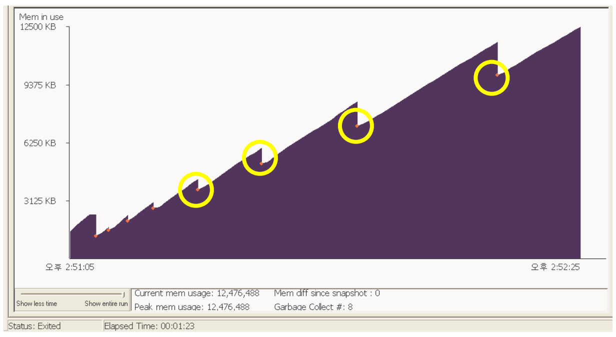




Assumption & Criteria

- Assumption
  - Source A and B provides same functionalities (e.g. B-tree, insert, sort, etc)
  - Same number of inputs (100,000 keys)
- Criteria
  - Compare memory and execution speed in rum time

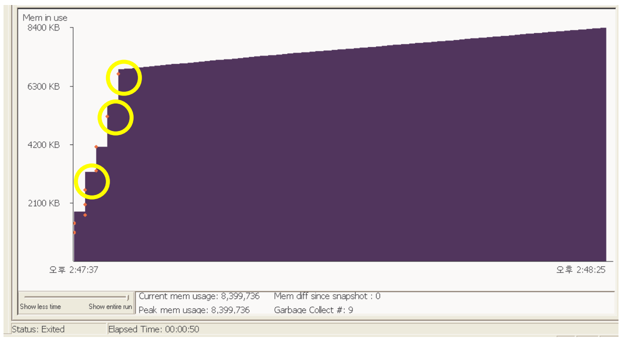
Source A (Memory Profile)



Current mem usage: 12,476,488 Mem diff since snapshot : 0
Peak mem usage: 12,476,488 Garbage Collect #: 8
Status: Exited Elapsed Time: 00:01:23

- Yellow Circle: Garbage Collection
- Memory consumption gradually increasing (ends with 12 mega bytes)
- Execution time (1min23sec) 🡐 Including Purify overhead

Source B (Memory Profile)



Current mem usage: 8,399,736   Mem diff since snapshot : 0
Peak mem usage: 8,399,736   Garbage Collect #: 9

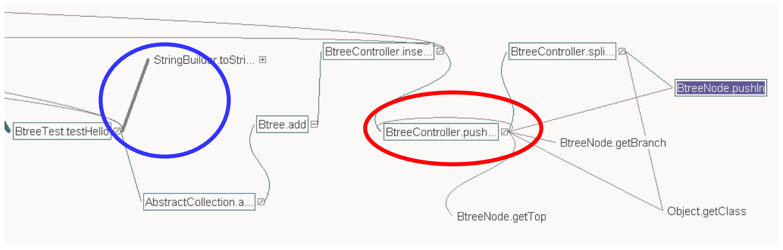
Status: Exited   Elapsed Time: 00:00:50

- Yellow Circle: Garbage Collection
- Memory consumption gradually increasing
  - But, not as high as Source A, ends with 8.4 mega bytes
- Execution time (50sec) 🡐 Including Purify overhead

Why Source B?

- Memory
  - Source A: Allocates more memory gradually by
  - Source B: Allocates most of memory when input keys are added into tree, however, less memory
- Execution time
  - Purify shows elapsed time including its overhead (Not true)
  - Without Purify, both have nearly same execution time (Source B is slightly fast, though)
- However, we would like to modify source B for better performance

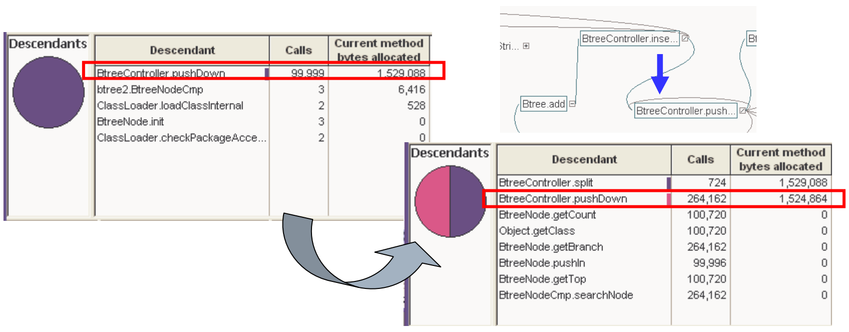
Source B Call Graph



- Injecting key takes most of Memory (blue circle, out of concern)
- Recursive structure to push keys into tree
  - Many calls and memories are allocated here

➔ *Eliminate recursive structure*

Source B Function Detail

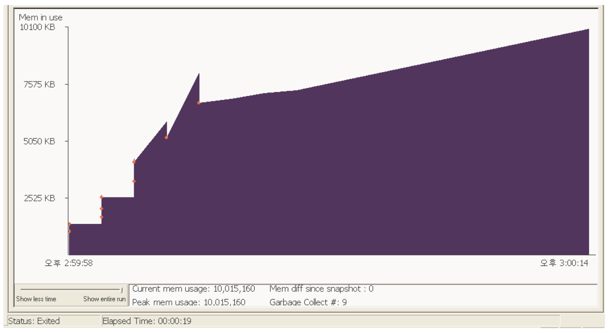


Descendant	Calls	Current method bytes allocated
BtreeController.pushDown	99,999	1,529,088
btree2.BtreeNodeCmp	3	6,416
ClassLoader.loadClassInternal	2	528
BtreeNode.init	3	0
ClassLoader.checkPackageAcce...	2	0

Descendant	Calls	Current method bytes allocated
BtreeController.split	724	1,529,088
BtreeController.pushDown	264,162	1,524,864
BtreeNode.getCount	100,720	0
Object.getClass	100,720	0
BtreeNode.getBranch	264,162	0
BtreeNode.pushIn	99,996	0
BtreeNode.getTop	100,720	0
BtreeNodeCmp.searchNode	264,162	0

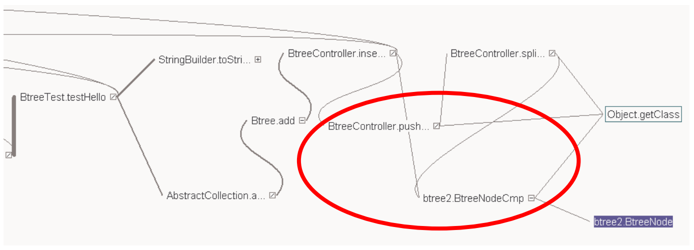
- Need same amount of memory for recursive call
- About 2.6 times call overhead with 1.5 megabytes additional memory allocation
- 1.5 mega bytes is not included in the entire memory consumption (8.4 mega bytes) ➔ Maybe Purify does not show stack memory?

Non-Recursive Source B (Memory Profile)



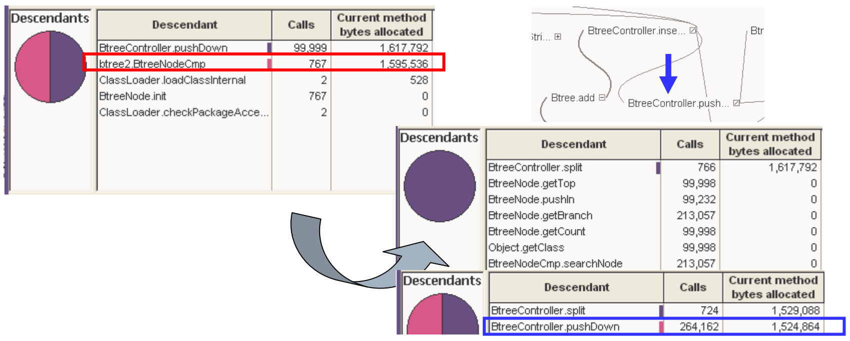
- More memory allocated due to additional TreeNode
  - Approx. 10 megabytes vs. approx. 8.4 mega bytes
- Execution time (19sec)
  - Surprising? 50 sec. vs. 19 sec. ➔ It's NOT pure execution time (Real execution time is twice faster than recursive.)

Non Recursive Call Graph



- No recursive call, but introduced additional TreeNode storage: Memory overhead

Call Graph (Recursive)



Descendants

Descendant	Calls	Current method bytes allocated
BtreeController.pushDown	99,999	1,617,792
btree2.BtreeNodeCmp	767	1,595,536
ClassLoader.loadClassInternal	2	528
BtreeNode.init	767	0
ClassLoader.checkPackageAcce...	2	0

Descendants

Descendant	Calls	Current method bytes allocated
BtreeController.split	766	1,617,792
BtreeNode.getTop	99,998	0
BtreeNode.pushIn	99,232	0
BtreeNode.getBranch	213,057	0
BtreeNode.getCount	99,998	0
Object.getClass	99,998	0
BtreeNodeCmp.searchNode	213,057	0

Descendants

Descendant	Calls	Current method bytes allocated
BtreeController.split	724	1,529,088
BtreeController.pushDown	264,162	1,524,864

- Additional TreeNode
  - approx 1.5 mega bytes (see, red rectangle)
- But, less call than recursive structure
  - It's much Faster (see, blue rectangle)

Application in Studio Project

- Analyze Studio Source Code with Purify
  - Check the memory usage and the memory leak point when handling the large data
  - Compare the performance with various algorithms for constructing tree traversing
- Opportunities for improvement
  - Find out memory consuming functions with call graph in Purify, re-code the functions, and continue to analyze until the performance is satisfactory
  - Help for making a decision of choosing an appropriate algorithm for our studio project
  - Coordinate the concurrent process: Measure the peak memory and avoid the peak memory usage with each other in concurrent process
  - Help to determine the minimum requirement resource to run the application

Benefit & Drawback of Purify

- Benefits
  - Clear overview of memory consumption in runtime
  - Number of call and allocated memory in each class
- Drawback
  - Purify can not detect memory access in Stack?
    - User have to see the function detail and calculate (No mention in the manual, though)
  - System resource to run Purify
    - Needs huge amount of memory
  - Irregular elapsed time (not proportional to execution speed)

Lattix LDM Tool Evaluation

Team OverHEAD
Karim Jamal & Clinton Jenkins

Tool Evaluation Project Presentation
17-654: Analysis of Software Artifacts

Outline

- Lattix LDM Tool Description
- Version of Tool
- Quantitative Data
- Case Study: A3 Project Description
- A3 Analysis Example
- Weaknesses
- Strengths
- Lessons Learned
- Questions?

Lattix LDM Tool Description

- Lightweight Dependency Modeler (LDM)
- Displays dependencies in a Design Structure Matrix (DSM) diagram
- Uses DSM partitioning algorithms to restructure diagram and identify logical subsystems
- Usable with Java and C/C++ projects

Version of Tool

- We evaluated the Community Version of the tool
- As compared to the Full Version, the Community Version:
  - Does not allow design rules to be specified
  - Does not enforce dependency constraints between different versions of a project
  - Does not expire
  - Is FREEEEE

Quantitative Data

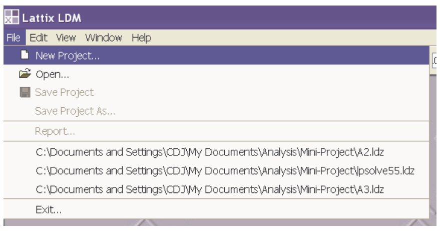
- Manually identified syntactic dependencies and then compared results to dependencies identified by tool.

Project Category	Project Name	Dependency Measurements	Count
Trivial	Trivial	Identified Correctly	0
		Failed to Identify	0
		Extraneously Identified	0
Mid-sized	A1	Identified Correctly	15
		Failed to Identify	0
		Extraneously Identified	0
	A2	Identified Correctly	16
		Failed to Identify	0
		Extraneously Identified	0
	A3	Identified Correctly	51
		Failed to Identify	0
		Extraneously Identified	0
	lpsolve	Identified Correctly	26
		Failed to Identify	0
		Extraneously Identified	0

Case Study: A3 Project Description

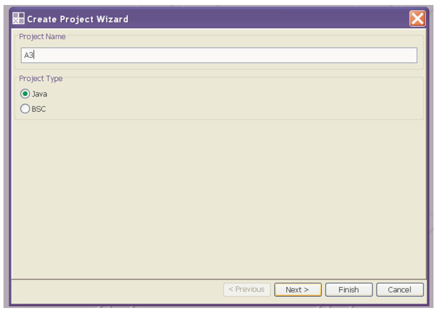
- Student scheduling application implemented for Architecture class
- Uses Implicit Invocation architectural style
- Components interact with each other by placing events onto an event bus and registering to receive events

A3 Analysis Example



C:\Documents and Settings\CDJ\My Documents\Analysis\Mini-Project\A2.ldz
C:\Documents and Settings\CDJ\My Documents\Analysis\Mini-Project\lpsolve55.ldz
C:\Documents and Settings\CDJ\My Documents\Analysis\Mini-Project\A3.ldz

1. Create a New Project

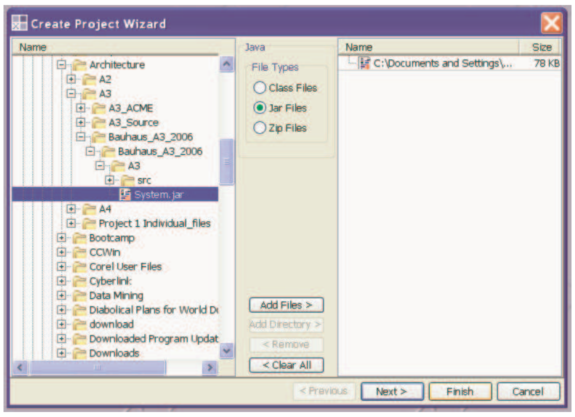


Project Name: A3

Project Type: [✓ Java] BSC

2. In this example we are dealing with a Java project, so select that option

A3 Analysis Example


Create Project Wizard

Name
Architecture
A2
A3
A3_ACME
A3_Source
Bauhaus_A3_2006
Bauhaus_A3_2006
A3
src
System.jar
A4
Project 1 Individual_files
Bootcamp
CCWin
Corel User Files
Cyberlink
Data Mining
Diabolical Plans for World Do
download
Downloaded Program Updat
Downloads

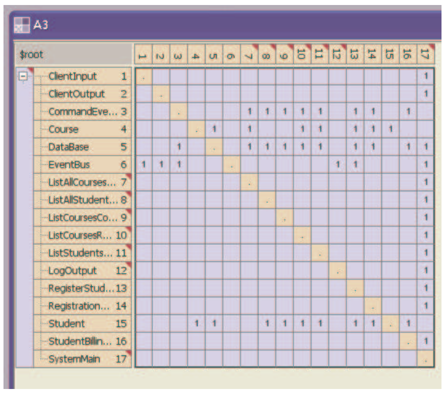
Java

File Types: ( ) Class Files [✓ Jar Files] ( ) Zip Files

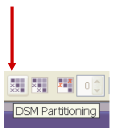
Name	Size
C:\Documents and Settings\...	78 KB

3. Select the .jar file in this case, but the .class files would have worked just as well. Hit Finish

A3 Analysis Example

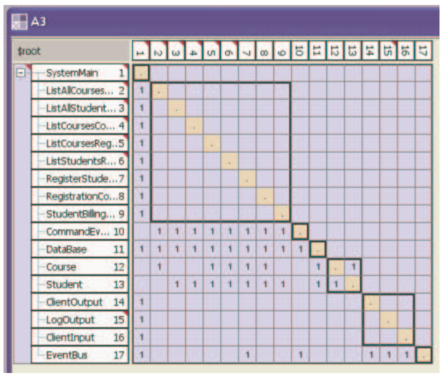

A3

$root		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
ClientInput	1	.																1
ClientOutput	2		.															1
CommandEve...	3			.				1	1	1	1	1		1	1		1	
Course	4				.	1		1			1	1		1	1	1		
DataBase	5			1		.		1	1	1	1	1		1	1		1	1
EventBus	6	1	1	1			.						1	1				1
ListAllCourses...	7							.										1
ListAllStudent...	8								.									1
ListCoursesCo...	9									.								1
ListCoursesR...	10										.							1
ListStudents...	11											.						1
LogOutput	12												.					1
RegisterStud...	13													.				1
Registration...	14														.			1
Student	15				1	1			1	1	1	1		1	1	.	1	
StudentBillin...	16																.	1
SystemMain	17																	.



4. The initial diagram presented for A3. Select all of the rows and push the DSM Partitioning button on the toolbar to rearrange the diagram.

A3 Analysis Example



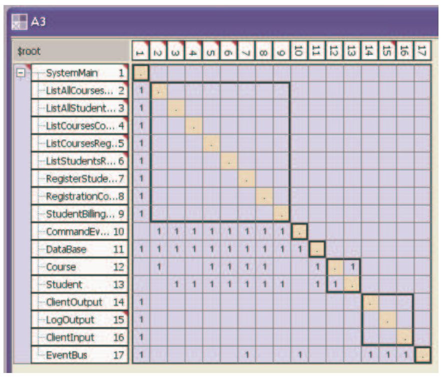
A3

$root		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
SystemMain	1	.																
ListAllCourses...	2	1	.															
ListAllStudent...	3	1		.														
ListCoursesCo...	4	1			.													
ListCoursesReg..	5	1				.												
ListStudentsR...	6	1					.											
RegisterStude...	7	1						.										
RegistrationCo...	8	1							.									
StudentBilling...	9	1								.								
CommandEv...	10		1	1	1	1	1	1	1	1	.							
DataBase	11	1	1	1	1	1	1	1	1	1	1	.						
Course	12		1			1	1	1	1			1	.	1				
Student	13			1	1	1	1	1	1	1		1	1	.				
ClientOutput	14	1													.			
LogOutput	15	1														.		
ClientInput	16	1															.	
EventBus	17	1						1			1				1	1	1	.

5. The tool rearranges the DSM diagram into a lower block-triangular form. The dark outline boxes identify logical subsystems within the project.

Weaknesses

- Only syntactic dependencies are identified.
- In the A3 case, there are semantic dependencies among rows 2-9 but the tool loses track of them when the CommandEventHandler and EventBus classes allow an indirect communication method.



A3

$root		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
SystemMain	1	.																
ListAllCourses...	2	1	.															
ListAllStudent...	3	1		.														
ListCoursesCo...	4	1			.													
ListCoursesReg..	5	1				.												
ListStudentsR...	6	1					.											
RegisterStude...	7	1						.										
RegistrationCo...	8	1							.									
StudentBilling...	9	1								.								
CommandEv...	10		1	1	1	1	1	1	1	1	.							
DataBase	11	1	1	1	1	1	1	1	1	1	1	.						
Course	12		1			1	1	1	1			1	.	1				
Student	13			1	1	1	1	1	1	1		1	1	.				
ClientOutput	14	1													.			
LogOutput	15	1														.		
ClientInput	16	1															.	
EventBus	17	1						1			1				1	1	1	.

Weaknesses

- Polymorphism was injected into A3 in the following fashion:

```
ListAllStudentsHandler objCommandEventHandler1 =
            new ListAllStudentsHandler(
            db,
            new
int[]{EventBus.EV_LIST_ALL_STUDENTS},
            EventBus.EV_SHOW);
.
.
.
```

Original Code

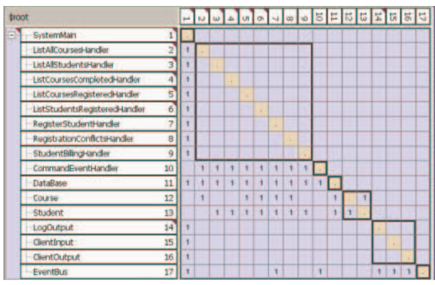
```
CommandEventHandler objCommandEventHandler1 =
            new ListAllStudentsHandler(
            db,
            new
int[]{EventBus.EV_LIST_ALL_STUDENTS},
            EventBus.EV_SHOW);
.
.
.
```

Code with Polymorphism Injected

Code is functionally identical but…

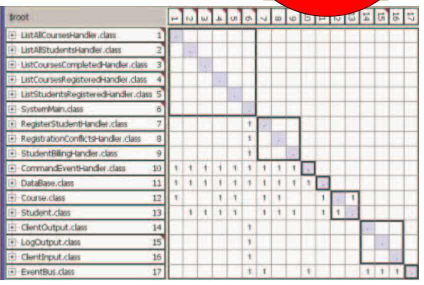
Weaknesses

- Lattix LDM identifies different dependencies with and without polymorphism.



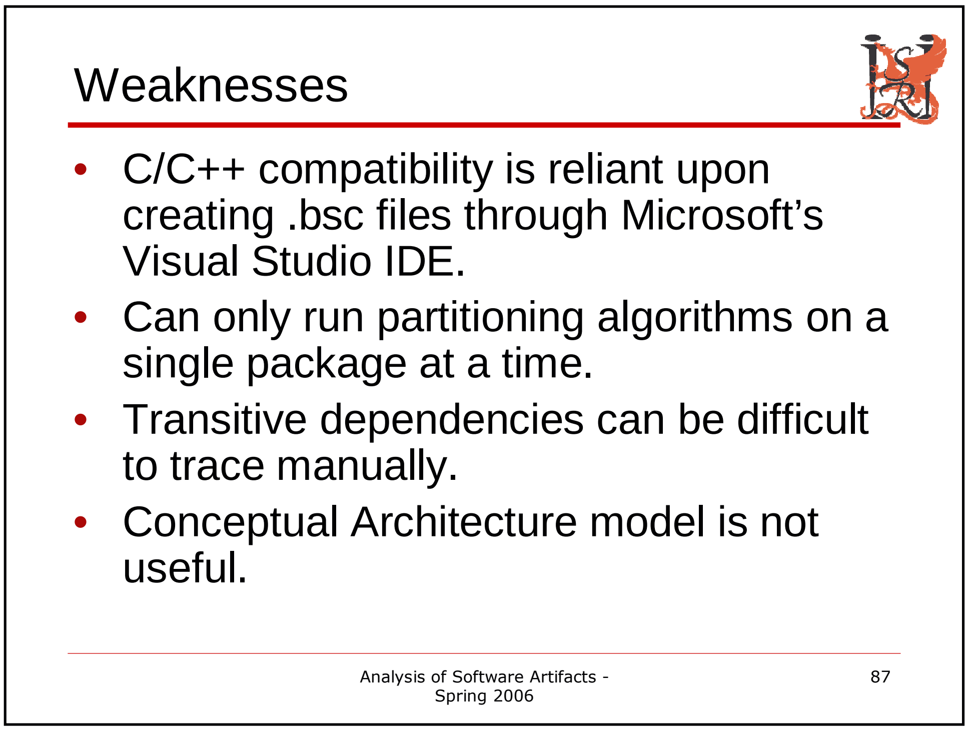
$root		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
SystemMain	1	.																
ListAllCoursesHandler	2	1	.															
ListAllStudentsHandler	3	1		.														
ListCoursesCompletedHandler	4	1			.													
ListCoursesRegisteredHandler	5	1				.												
ListStudentsRegisteredHandler	6	1					.											
RegisterStudentHandler	7	1						.										
RegistrationConflictsHandler	8	1							.									
StudentBillingHandler	9	1								.								
CommandEventHandler	10		1	1	1	1	1	1	1	1	.							
DataBase	11	1	1	1	1	1	1	1	1	1	1	.						
Course	12		1			1	1	1	1			1	.	1				
Student	13			1	1	1	1	1	1	1		1	1	.				
LogOutput	14	1													.			
ClientInput	15	1														.		
ClientOutput	16	1															.	
EventBus	17	1						1			1				1	1	1	.

Original system



$root		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
ListAllCoursesHandler.class	1	.																
ListAllStudentsHandler.class	2		.															
ListCoursesCompletedHandler.class	3			.														
ListCoursesRegisteredHandler.class	4				.													
ListStudentsRegisteredHandler.class	5					.												
SystemMain.class	6						.											
RegisterStudentHandler.class	7						1	.										
RegistrationConflictsHandler.class	8						1		.									
StudentBillingHandler.class	9						1			.								
CommandEventHandler.class	10	1	1	1	1	1	1	1	1	1	.							
DataBase.class	11	1	1	1	1	1	1	1	1	1	1	.						
Course.class	12	1			1	1		1	1			1	.	1				
Student.class	13		1	1	1	1		1	1	1		1	1	.				
ClientOutput.class	14						1								.			
LogOutput.class	15						1									.		
ClientInput.class	16						1										.	
EventBus.class	17						1	1			1				1	1	1	.

System with polymorphism

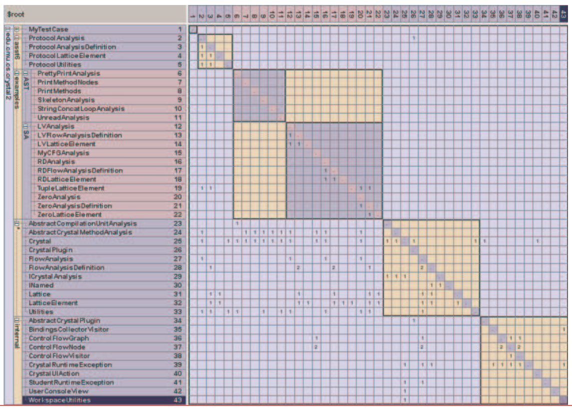


Weaknesses

- C/C++ compatibility is reliant upon creating .bsc files through Microsoft’s Visual Studio IDE.
- Can only run partitioning algorithms on a single package at a time.
- Transitive dependencies can be difficult to trace manually.
- Conceptual Architecture model is not useful.

Strengths

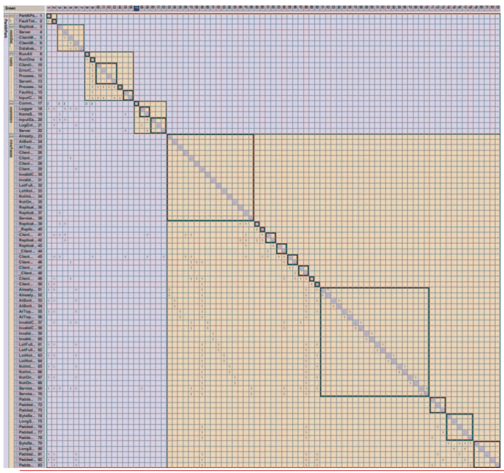
- Hierarchical support among packages is great for abstracting away extra information not needed at the time.



MyTestCase 1
ProtocolAnalysis 2
ProtocolAnalysisDefinition 3
ProtocolLatticeElement 4
ProtocolUtilities 5
PrettyPrintAnalysis 6
PrintMethodNodes 7
PrintMethods 8
SkeletonAnalysis 9
StringConcatLoopAnalysis 10
UnreadAnalysis 11
LVAnalysis 12
LVFlowAnalysisDefinition 13
LVLatticeElement 14
MyCFGAnalysis 15
RDAnalysis 16
RDFlowAnalysisDefinition 17
RDLatticeElement 18
TupleLatticeElement 19
ZeroAnalysis 20
ZeroAnalysisDefinition 21
ZeroLatticeElement 22
AbstractCompilationUnitAnalysis 23
AbstractCrystalMethodAnalysis 24
Crystal 25
CrystalPlugin 26
FlowAnalysis 27
FlowAnalysisDefinition 28
ICrystalAnalysis 29
INamed 30
Lattice 31
LatticeElement 32
Utilities 33
AbstractCrystalPlugin 34
BindingsCollectorVisitor 35
ControlFlowGraph 36
ControlFlowNode 37
ControlFlowVisitor 38
CrystalRuntimeException 39
CrystalUIAction 40
StudentRuntimeException 41
UserConsoleView 42
WorkspaceUtilities 43

Strengths

- Tool is fast and seems to scale well.

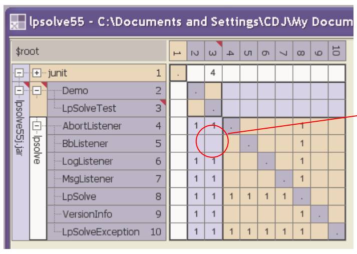


Project Category	Project Name	SLOC
Trivial	Trivial	15
Mid-sized	A1	369
	A2	530
	A3	684
	Ipsolve	509
Large, complex	Crystal2	4244
	ParkNPark	6466

DSM for ParkNPark, a 6466 SLOC project.

Strengths

- You can cross-check a project against its test files as another method to ensure all classes are being tested.



lpsolve55 - C:\Documents and Settings\CDJ\My Docum

$root		1	2	3	4	5	6	7	8	9	10
junit	1	.		4							
Demo	2		.								
LpSolveTest	3			.							
AbortListener	4		1	1	.				1		
BbListener	5					.			1		
LogListener	6		1	1			.		1		
MsgListener	7		1	1				.	1		
LpSolve	8		1	1	1	1	1	1	.		
VersionInfo	9		1	1					1	.	
LpSolveException	10		1	1	1	1	1	1	1		.

LpSolveTest is a set of unit tests; it never uses BbListener, indicating a possible hole in the test suite.

Lessons Learned

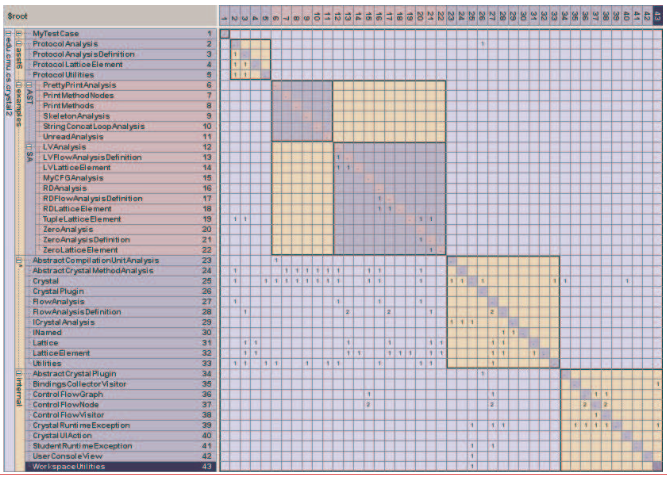
- Lattix LDM is a great place to start for architectural discovery and a good way to track dependencies.
- Systems with many semantic dependencies and few syntactic dependencies will be difficult to work with in Lattix LDM. This can still be overcome with manual marking of dependencies though.
- Is based upon DSM technology, which is continuing to mature; this will also allow the tool to take advantage of new, clever DSM partitioning algorithms that may be invented in the future.

Questions? …we all have them

- No, this will not spit out the notional architecture for your project
- Yes, this does also come as an Eclipse plug-in (what doesn't nowadays)
- No, we don't know why the DSMs produced by the Eclipse plug-in are more colorful than the ones produced by the stand-alone application
- Yes, we did run it against the Cystal2 project…

So how did Crystal2 fare?

- Relatively well



```
$root
edu.cmu.cs.crystal2
  assf6
    MyTestCase                          1
    ProtocolAnalysis                    2
    ProtocolAnalysisDefinition          3
    ProtocolLatticeElement              4
    ProtocolUtilities                   5
  examples
    AST
      PrettyPrintAnalysis               6
      PrintMethodNodes                  7
      PrintMethods                      8
      SkeletonAnalysis                  9
      StringConcatLoopAnalysis         10
      UnreadAnalysis                   11
    SA
      LVAnalysis                       12
      LVFlowAnalysisDefinition         13
      LVLatticeElement                 14
      MyCFGAnalysis                    15
      RDAnalysis                       16
      RDFlowAnalysisDefinition         17
      RDLatticeElement                 18
      TupleLatticeElement              19
      ZeroAnalysis                     20
      ZeroAnalysisDefinition           21
      ZeroLatticeElement               22
  *
    AbstractCompilationUnitAnalysis    23
    AbstractCrystalMethodAnalysis      24
    Crystal                            25
    CrystalPlugin                      26
    FlowAnalysis                       27
    FlowAnalysisDefinition             28
    ICrystalAnalysis                   29
    INamed                             30
    Lattice                            31
    LatticeElement                     32
    Utilities                          33
  internal
    AbstractCrystalPlugin              34
    BindingsCollectorVisitor           35
    ControlFlowGraph                   36
    ControlFlowNode                    37
    ControlFlowVisitor                 38
    CrystalRuntimeException            39
    CrystalUIAction                    40
    StudentRuntimeException            41
    UserConsoleView                    42
    WorkspaceUtilities                 43
```